SConE: Simplified Cone Embeddings with Symbolic Operators for Complex Logical Queries

Chau Duc Minh Nguyen and Tim French and Wei Liu and Michael Stewart

ARC Centre for Transforming Maintenance through Data Science,
Department of Computer Science and Software Engineering
School of Physics, Mathematics and Computing
The University of Western Australia
chau.nguyenducminh@research.uwa.edu.au
{tim.french, wei.liu, michael.stewart}@uwa.edu.au

Abstract

Geometric representation of query embeddings (using points, particles, rectangles and cones) can effectively achieve the task of answering complex logical queries expressed in first-order logic (FOL) form over knowledge graphs, allowing intuitive encodings. However, current geometric-based methods depend on the neural approach to model FOL operators (conjunction, disjunction and negation), which are not easily explainable with considerable computation cost. We overcome this challenge by introducing a symbolic modeling approach for the FOL operators, emphasizing the direct calculation of the intersection between geometric shapes, particularly sector-cones in the embedding space, to model the conjunction operator. This approach reduces the computation cost as a non-neural approach is involved in the core logic operators. Moreover, we propose to accelerate the learning in the relation projection operator using the neural approach to emphasize the essential role of this operator in all query structures. Although empirical evidence for explainability is challenging, our approach demonstrates a significant improvement in answering complex logical queries (both non-negative and negative FOL forms) over previous geometric-based models.

1 Introduction

Answering complex logical queries is a fundamental task of knowledge graphs (KGs) (Bollacker et al., 2008; Vrandečić and Krötzsch, 2014; Speer et al., 2017; Fellbaum, 2010; Lehmann et al., 2015; Mitchell et al., 2018) for various purposes of individuals and businesses. Conventional methods, such as Hartig and Heese (2007); Schmidt et al. (2010), have been well-studied on complete KGs. However, these methods face challenges in incomplete and largely-scaled KGs, as conventional methods cannot traverse graphs via missing connections. Time complexity is another challenge as it grows exponentially during the traversal process. Modern approaches, such as Hamilton et al. (2018); Ren et al. (2020), use query embeddings (QEs) methods that can answer complex logical queries without the need of path traversal in graphs. The QEs methods first transform a complex logical query into a machine-readable format: (1) converting a query in natural textual form into first-order logic (FOL) form (including conjunction $\land$, disjunction $\lor$, negation $\neg$ and existential quantification $\exists$ operator) and (2) decomposing it into a computation graph (including relation projection operator). For example, Fig. 1 depicts the process of turning a complex logical query “Which universities do the Nobel Prize winners of Australian citizens work in?” into a computation graph. This FOL query is then projected in the embedding space, required for the modeling process to learn to answer the query.

Among different approaches in representing queries in the embedding space, geometric-based approaches have had renewed interests since the work in point embeddings (Hamilton et al., 2018). Following works have expanded this approach using hyper-boxes (Ren et al., 2020), sets of points as particles (Bai et al., 2022), hyperboloids (Choudhary et al., 2021b) and 2D-cones (Zhang et al., 2021). These works commonly resort to set operators over shapes that can handle the conjunction, only a few Zhang et al. (2021) can handle the negation. Nevertheless, existing geometric-based methods depend on the neural approach to model the conjunction operator. This approach is not easily explainable, counter-intuitive and does not take full advantage of the properties that these geometric representations are intended to be used for.

We highlight in this paper the essential role the projection operator plays in all complex query embedding methods. This operator is often learned through training neural architectures together with the logical operators in an end-to-end fashion. The semantic role of this operator is to obtain a meaningful representation of a predicate (relation) in
FOL, which operates as a function converting a domain of input embeddings into a range of outputs. These are fundamentally different from the roles of the logical operators, in which geometric approaches can be modelled as set operators. Moreover, little work highlights the importance of the neural approach in learning the relation projection.

We address the first issue by introducing a novel symbolic operator in geometry for answering the conjunctive queries. Specifically, we directly calculate the intersection of geometric shapes in the embedding space (see an example in Fig. 2 and more details in Sec. 4.2). We use cone embeddings (Zhang et al., 2021) as a key geometric representation in our approach, since conic shapes were shown to be effective in modeling all FOL operators. By directly calculating intersection, our approach can reduce the computational cost of modeling the conjunction operator, as there is no need to incorporate expensive neural training in this logic operator, as compared with other geometric-based models (Hamilton et al., 2018; Ren et al., 2020; Choudhary et al., 2021b; Zhang et al., 2021). Further, classifying types of geometric intersection (partial, complete and none type) can improve the explainability in modeling conjunction operator (see Sec 4.2). To highlight the importance of relation projection (finding tail entities from a source entity via a relation) in complex logical query embeddings, we propose a general framework of modeling this operator, called relation projection network (RPN) (see Fig. 3). The RPN can enhance the learning in the relation projection operation, due to its high frequency and its dominance in diverse query structures (see Fig. 6).

Overall, we introduce Simplified Cone Embeddings (SConE) for modeling the relation projection and logical operators in complex queries. Our contributions are: (1) introducing a symbolic modeling for the conjunction operator in FOL query, (2) proposing a general framework using RPN to improve the learning of relation projection operator in both atomic and complex logical queries and (3) surpassing model performance of previous state-of-the-art geometric-based models for both non-negation and negation queries.

2 Related Work

Atomic query answering for knowledge graph completion The atomic query has a given head concept \(v_h\) and a relation \(r\), and the answering task is to find the projected tail concept \(v_t\). Using geometric-based methods to answer atomic queries (or path queries) without complex logical operators has been well-studied since the appearance of knowledge graph embeddings, notably, translation-based methods (Bordes et al., 2013) and rotation (Sun et al., 2019; Zhang et al., 2020a). Further, Nickel and Kiela (2017); Balažević et al. (2019) proposed hyperbolic space (non-Euclidean geometry) over a Poincaré ball while others (Gao et al., 2020) used 3D shapes. However, these models are limited in answering complex queries involving FOL logical operators (e.g. Fig. 1 and Fig. 6).

Complex logical query answering for multi-hop reasoning The complex logical query has atomic queries with logical operators (see Fig. 1). Different methods addressing this task are geometry-based embeddings (points Hamilton et al. (2018), boxes Ren et al. (2020), hyperboloids Choudhary et al. (2021b), cones Zhang et al. (2021), particles Bai et al. (2022)), distribution-based embeddings (Ren and Leskovec, 2020; Choudhary et al., 2021a; Huang et al., 2022; Yang et al., 2022; Long et al., 2022), auxiliary enrichment methods (Hu et al., 2022) using entity and relation type knowledge, logic-based methods (Arakelyan et al., 2021; Chen et al., 2022; Zhu et al., 2022; Xu et al., 2022) using fuzzy logic to model the logical operators, neural-based methods (Kotnis et al., 2021; Liu et al., 2022; Amayuelas et al., 2022) and oth-
ers (Sun et al., 2020). Although the logic-based methods are explainable in modeling FOL operators (a learning-free), other methods such as geometric-based embeddings are challenging to interpret this process, since these rely on the neural approach to model the conjunction operator. We provide a symbolic modeling approach for handling the conjunction to improve explainability in geometric-based models.

3 Preliminaries

3.1 First-Order Logic queries over KGs

Given a set of entities \( v \in V \) and a set of relations \( r \in R \), a knowledge graph (KG) \( G = \{(v_h, r, v_t)\} \) is a set of triples, each includes a head entity \( v_h \), a relation \( r \) and a tail entity \( v_t \).

Given a knowledge graph, a complex FOL query is a formula consisting of: constants, quantified bound variables \( (V_1, \ldots, V_n) \) and free variables \( (V'_2) \) (target), in addition to relation symbols \( R(V_i, V_j) \) and logic connectives \( (\exists, \land, \lor, \neg) \). An entity of KG \( v \in V \) maps to each constant or variable. Each \( R(V_i, V_j) \) maps to a binary function whether a relation exists between \( V_i \) and \( V_j \). Logic connectives are conjunction \( (\land) \), disjunction \( (\lor) \), negation \( (\neg) \) and existential quantification \( (\exists) \) (see an example of FOL query mapped to the (ip) structure in Fig. 1, and more query structures in Fig. 6). Given this example, the goal of a FOL query answering is to find the answers (or free variables) such that the formula is true.

3.2 Query Embeddings

Cone parameterization. We adopt the same definitions and propositions in Zhang et al. (2021), to define a two-dimensional sector-cone using two variables: (1) angle \( \alpha \in [-\pi, \pi) \) represents the angle between the semantic center axis and the positive \( x \)-axis, and (2) aperture \( \beta \in [0, 2\pi] \) represents the aperture of the sector cone (see an example with pink sector-cone in Fig. 2).

Query Embeddings representation. Given a complex logical query \( q \), we represent its embedding \( (q) \) as a Cartesian product of two-dimensional sector-cones in the embedding space using two variables: semantic center axis \( \alpha_q \in [-\pi, \pi)^d \) and aperture \( \beta_q \in [0, 2\pi]^d \), where \( d \) is the embedding dimension. Next, given a semantic entity \( v \), we represent its embedding \( (v) \) as a Cartesian product of cones embedding using semantic center axis \( \alpha_v \in [-\pi, \pi)^d \) and zero aperture defined by:

\[
q = (\alpha_q, \beta_q), \ v = (\alpha_v, 0) \tag{3.1}
\]

3.3 Operators in First-Order Logic queries

We decompose the symbolic representation of the complex logical query \( (q) \) using a computation graph, a tree-like query (see Fig. 1). This graph has vertexes and links where each vertex represents a set of entities and each link represents a modeling process of either of two types: relation projection operator \( (P) \) or any FOL operators (conjunction \( (C) \), disjunction \( (D) \) and negation \( (N) \)):

- **Relation Projection (Projection)**: \( P(x, r) \) computes the projection from the input \( x \) as a head entity to the set of tail entities via relation \( r \). Otherwise, \( P(x, r^{-1}) \) computes the projection from the input \( x \) as a tail entity to the set of head entities via \( r \).

- **Conjunction (Intersection)**: \( C(x_1, x_2) \) computes the intersection of each geometric element in one set of entities \( (x_1) \) and the corresponding element in the other entity set \( (x_2) \).
• **Disjunction (Union):** $D(x_1, x_2)$ computes the union of each geometric element in one set of entities ($x_1$) and the corresponding geometric element in the other entity set ($x_2$).

• **Negation (Complement):** $N(x)$ computes the complement of each geometric element in the set of entities ($x$).

4 Modeling operators of FOL queries

We describe the modeling of relation projection (4.1) and logical operators (4.2) in a complex FOL query ($q$) (with its set of answer entities $V_q \subset V$) over knowledge graphs in the following:

4.1 Modeling relation projection

This section is to model the relation projection operator ($P$) (see Sec. 3.3) for Knowledge Graph Embedding (KGE). Overall, given an atomic query $q = (v, r)$, we propose a relation projection network (RPN) with two layers: (1) first transforming the source entity using an ensemble of multiple KGE techniques, (2) merging the outputs at the second layer (called entanglement layer) to produce the sector-cones embedding of the query (see Fig. 3). We use two KGE techniques at the first layer, called relation transformation and multi-layer perceptron. As the relation projection task is similar to KGE, one can select different models in KGE such as TransE (Bordes et al., 2013) or HAKE (Zhang et al., 2020b), then adapt these into the first layer in principle.

![Figure 3: (Up→Down): A general framework of RPN.](image)

There are many models achieving KGE (Ji et al., 2022), the ensemble selection is therefore not restricted, but it should be efficient in model complexity and computational cost. Fundamentally, using one technique is sufficient; however, having a general framework using multiple techniques is to analyze the learning process from a broader viewpoint. We select two KGE models as a simple case to illustrate that it is possible to use multiple KGE techniques.

**Relation transformation** Specifically, we model an embedded relation $r = (W_r, b_r)$ requiring for the projection operation ($P$) by a neural network as in (Chen et al., 2022), where $W_r$ denotes a weight matrix and ($b_r$) denotes a bias vector. We transform a source entity $v = (\alpha_v, \beta_v)$ into an embedded query ($q$) via this relation. However, as our entity representation based on sector-cones embeddings, which is different than the fuzzy sets used in (Chen et al., 2022), we add a concatenation operation of the semantic center axis ($\alpha_v$) and the aperture ($\beta_v$) to convert these into a vector $[v] \in \mathbb{R}^{2d}$ as follows:

$$q_t = f(v) = LN(W_r[v] + b_r),$$

where LN is Layer Normalization (Ba et al., 2016). We use the basic decomposition of (Schlichtkrull et al., 2018) to define $(W_r)$ and $(b_r)$.

**Multi-layer Perceptron (MLP)** An alternative way to model the relation projection is to use MLP. We transform the entity ($v$) to query ($q$) via the relation ($r$) by a mapping function ($f$) as follows:

$$q_m = f(v_r) = LN(MLP([v_r])),$$

where MLP : $\mathbb{R}^{2d} \rightarrow \mathbb{R}^{2d}$ is to approximately represent the mapping function $f(x)$, $v_r$ is a translation embeddings of the source entity and the relation: $v_r = v + r$. As the representation of the entity and the relation $r = (\alpha_r, \beta_r)$ are sector-cones embedding, we apply a concatenation operation as that in the relation transformation technique to convert ($v_r$) to the vector embedding $[v_r] \in \mathbb{R}^{2d}$.

**Entanglement layer** After transforming the entity to the embedded query using the relation transformation and the MLP, we introduce an entanglement layer to merge the output from the first KGE layer into one output. We use attention mechanism in this layer:

$$q = (\alpha_q, \beta_q) = s \left( \sum_{i=1}^{2d} A \odot [q_t, q_m] \right),$$

where $s(x)$ is a function to split the $2d$-vector into two $d$-vectors, each is for the semantic center axis and the aperture embedding of the query, $\odot$ denotes Hadamard product, $[\cdot]$ denotes an operator to stack two $d$-vectors into a matrix in $\mathbb{R}^{2 \times 2d}$ and $A \in \mathbb{R}^{2 \times 2d}$ is an attention matrix as follows:

$$A = \text{SoftMax} \left( f_a(q_t, q_m) \right),$$

11934
where the SoftMax(.) function applies over the first dimension of the matrix. The \( f_a(q_t, q_m) = \text{MLP}([q_t, q_m]) \) is attention score function. We also provide a scaling function to convert the semantic center axis \((\alpha_q)\) and the aperture \((\beta_q)\) into their normal range (see Appendix B.1) as that in ConE (Zhang et al., 2021).

4.2 Symbolic modeling of logical operators

In this section, we describe the modeling process of all logical operators \((C, N, D)\) using symbolic modeling only without neural-based methods, naturally making use of the geometric properties of sector-cone shapes in the embedding space. In comparison with ConE (Zhang et al., 2021), this model leveraged the neural approach to learn the conjunction \((C)\) while using non-neural approach to model the disjunction and negation \((D, N)\).

Conjunction  This aims to model the conjunction \(C(q_1, q_2)\) of any pair of conjunctive queries, each query \(q_i = (\alpha_i, \beta_i)\) is in the cone embedding space. Assuming the embedding dimension \((d = 1)\) as the simplest case, each query is represented by a sector-cone (see Eq. (3.1)). As the intersection of the two conjunctive queries is also a sector-cone \(q_\land = (\alpha_\land, \beta_\land)\); therefore, one can directly calculate this intersection from a symbolic geometric perspective as follows:

\[
\begin{align*}
\beta_\land & = \begin{cases} 
  u_2 - l_1, & \text{if } c_1 \\
  \beta_2, & \text{if } c_2 \\
  0, & \text{if } c_3 \\
  u_1 - l_2, & \text{if } c_4 \\
  \beta_1, & \text{if } c_5 \\
  0, & \text{if } c_6 
\end{cases} \\
\alpha_\land & = \begin{cases} 
  u_2 - \frac{\beta_\land}{2}, & \text{if } c_1 \\
  \alpha_2, & \text{if } c_2 \\
  l_1 - \frac{u_1 - u_2}{2}, & \text{if } c_3 \\
  u_1 - \frac{\beta_\land}{2}, & \text{if } c_4 \\
  \alpha_1, & \text{if } c_5 \\
  l_2 - \frac{l_2 - u_1}{2}, & \text{if } c_6 
\end{cases}
\end{align*}
\]

(4.1)

where \((u_i, l_i)\) is the upper and lower bound for each sector-cone \((l_i \leq \alpha_i \leq u_i)\). These calculations are that \((u_i = \alpha_i + \frac{\beta_i}{2})\) and \((l_i = \alpha_i - \frac{\beta_i}{2})\); and \(c_i\) represents each conditional scenario regarding relative position between the two sector-cones:

\[
\begin{align*}
c_1 & := (u_1 \geq u_2) \land (u_2 \geq l_1) \land (l_1 \geq l_2), \\
c_2 & := (u_1 \geq u_2) \land (u_2 \geq l_2) \land (l_2 > l_1), \\
c_3 & := (u_1 \geq l_1) \land (l_1 > u_2) \land (u_2 \geq l_2), \\
c_4 & := (u_2 \geq u_1) \land (u_1 \geq l_2) \land (l_2 \geq l_1), \\
c_5 & := (u_2 \geq u_1) \land (u_1 \geq l_1) \land (l_1 > l_2), \\
c_6 & := (u_1 \geq l_2) \land (l_2 > u_1) \land (u_1 \geq l_1).
\end{align*}
\]

(4.2)

Note that there are three types regarding calculating intersection of two sector-cones in Eq. (4.2): (1) partial intersection (see \(c_1, c_4\)), (2) complete intersection (see \(c_2, c_5\)) and (3) none intersection (see \(c_3, c_6\)). Figure 4 shows these cases \((c_1, c_2, c_3)\) from one sector-cone to the other and vice versa \((c_4, c_5, c_6)\). While the calculation of the partial and complete intersection are based on natural representation of geometric shapes, the calculation of none intersection type is based on zero aperture and middle semantic axis between the lower bound of one sector-cone and the upper bound of the other. The aperture in this situation is confidence, but the semantic axis is uncertain as it can be any axes between the mentioned bounds. We consider the middle axis as a special case for none intersection type (see further details in the following Eq. 4.3).

In general, to compute the intersection of \((k)\) conjunctive queries, assuming this computation satisfies the associative and/or commutative law for logic, we compute the intersection \(C(q_i, q_{i+1})\) of the first two arbitrary conjunctive queries, then compute the intersection of \(C(q_i, q_{i+1})\) and the next conjunctive query \((q_{i+2})\) to produce \(C(q_i, q_{i+1}, q_{i+2})\), and iterate this process until reaching the final conjunctive query \((q_k)\).

Conjunction: Weight semantic axis for none type intersection  In the conjunction \(C(q_1, q_2)\) with none type intersection \((c_3, c_4)\), the equality of axis intersection \((\alpha_\land)\) of two sector-cones embeddings can be any axes between the upper bound of one sector-cone and the lower bound of the other sector-cone. In general, the equality to calculate \((\alpha_\land)\) in the cone embedding space (see Eq. 4.1) for
the none type intersection is shown below:

\[
\alpha_\land = \begin{cases} 
\delta l_1 + (1-\delta)u_2, & \text{if } c_3, \\
\delta l_2 + (1-\delta)u_1, & \text{if } c_6,
\end{cases} \tag{4.3}
\]

where \(\delta \in [0, 1]\) is a hyper-parameter to control the spatial location of axis intersection regarding that of the two mentioned bounds. Notice that, let \(\delta = 0.5\) as in Eq. (4.1), the semantic center axis intersection \((\alpha_\land)\) is in the middle between the two bounds as a special case of equality in Eq. (4.3) (see Appendix F for further analysis).

**Negation** This aims to model the negation \(\neg N(q)\), called \(q_- = (\alpha_-, \beta_-)\) of the embedded query \(q = (\alpha, \beta)\). In the cone embedding space, the semantic center axis of \((q_-)\) should be in opposite direction via the \(O-\)axis regarding that of \((q)\) (see Fig. 5 (c)). In terms of the aperture, the summation of both apertures of \((q_-)\) and \(q\) should be close to \((2\pi)\) as follows:

\[
\alpha_- = \begin{cases} 
\alpha - \pi, & \text{if } (\alpha \geq 0) \\
\alpha + \pi, & \text{if } (\alpha < 0)
\end{cases} \quad \beta_- = 2\pi - \beta.
\]

**Disjunction** Similar to cone embedding Zhang et al. (2021), we adapt the DNF technique in Ren et al. (2020) to represent the disjunction operation \(D(q_1, q_2)\) as disjunction of conjunctive queries (see Fig 5 (b)). Hence, we can leverage \((C, N)\) operators above to have a set of embeddings of the conjunctive queries. Those entities nearest to any of these conjunctive queries in the cone embedding space are considered to be the answers (see the aggregated distance score in Eq. (4.5)).

### 4.3 Optimization

**Distance score function** We define a distance score function \(d(v, q)\) of the embedding between: the expected entity \(v = (\alpha, 0)\) and the query \(q = (\alpha_q, \beta_q)\) (as stated in Sec. 3.2). We use two distance types: \(d_{con}\) for conjunctive queries and \(d_{dis}\) for disjunctive queries as (Ren et al., 2020; Zhang et al., 2021). In \(d_{con}\), there are three terms: an outside distance \((d_o)\), an inside distance \((d_i)\) and a separated axis distance \((d_a)\) (see Fig. 5 (d)), which are defined by:

\[
d_{con}(v, q) = (1-\psi)(d_o + \lambda d_i) + \psi d_a, \tag{4.4}
\]

where \(\lambda \in (0, 1)\) is to encourage \((v)\) to be covered by the sector-cone embedding \((q)\). The hyper-parameter \((\psi)\) is to weight the effect of the outside vs inside distance and the separated axis distance (see Appendix C for more details). To calculate \((d_{dis})\), we use the DNF technique in Ren et al. (2020), which obtains the minimum distance in embeddings between: an expected entity and each conjunctive query in DNF, over a \((k)\) number of conjunctive queries:

\[
d_{dis}(v, q) = \min\{d_{con}(v, q_i)_{i:1\rightarrow k}\}, \tag{4.5}
\]

**Loss function** During the optimization process, we use the negative sampling loss \((L)\) (Mikolov et al., 2013a,b) as that in Ren and Leskovec (2020): \(L = L_1 + L_2\), where \(L_1 = -\log \sigma(\gamma - d(v, q))\) involves a minimization of the distance \((d(v, q))\) for a positive answer entity \((v \in \mathcal{V}_q)\), and \(L_2 = -\frac{1}{n}\sum n \log \sigma(d(v', q) - \gamma)\) involves a maximization of the distance \((d(v', q))\) for a number \((n)\) of negative answer entities \((v'_r \notin \mathcal{V}_q); \sigma(x)\) is the activation function (e.g. sigmoid) and \((\gamma)\) is a positive margin as hyper-parameter.

### 5 Experiments

#### 5.1 Experimental setups

**Multi-hop Reasoning (MHR) or Complex Logical Query Answering task** Given an arbitrary complex FOL query, when traversing the incomplete KGs, non-trivial answers cannot be returned directly. The MHR task aims to find these answers. We evaluate our approach on there datasets: FB15k (Bollacker et al., 2008), FB15k-237 (Toutanova and Chen, 2015) and NELL995 (Xiong et al., 2017), following the pre-processing in BetaE (Ren and Leskovec, 2020). We follow the training protocol of previous works (Ren and Leskovec, 2020; Zhang et al., 2021), using 10 query syntaxes (non-negation 1p/2p/3p/2i/3i and negation 2in/3in/inp/pin/pi) for the training. We use these 10 syntaxes plus 4 unseen syntaxes (ip/up/2u/pi) for the evaluating process (see Fig. 6). An example of the (1p) query is \((v, r_1)\) i.e. (Wesleyan_University, major_field_of_study), while (2p) or (3p) query corresponds to \((v, r_1, r_2)\) or \((v, r_1, r_2, r_3)\).

**Evaluation Protocol** Following the evaluation protocol in (Ren et al., 2020), given a query, we split its answers into two sets: easy answers and hard answers. The former is for those entities that can be reached on the training/validation graph through symbolic approach in graph traversing. The latter is for those that can be predicted using query embedding models, or the reasoning process
performs on hard answers. We use the mean reciprocal rank (MRR) metrics, computing the ranking of each hard answer against all non-answer entities, to measure the performance of models.

**Baselines** We use four recent geometric-based embedding models as baselines: GQE (Hamilton et al., 2018), Query2Box (Q2B) (Ren et al., 2020), Query2Particles (Q2P) (Bai et al., 2022) and ConE (Zhang et al., 2021), and obtain their results from ConE and Q2P. We also compare these results with state-of-the-art models based on fuzzy logic (see Appendix E.2).

### 5.2 Results

**Existential Positive First-order (EPFO) queries**

Overall, the average MRR in all EPFO queries without negation (AVG_p) of SConE significantly outperform all geometric-based baselines using the three datasets, particularly more than that of ConE by nearly 12% using the NELL995 dataset (see Table 1). For queries (1p/2p/3p/2i/3i) involving in the training process, most of the average MRR in each of these query structure (11 out of 15 metrics) significantly surpass baselines. Specially, in the (2p) query, around 26% gain of the MRR in SConE over that in ConE observes in the NELL995 dataset. With regard to queries (ip/pi/2u/up) that are not involved in the training process, the model performance of SConE also shows a significant increase of MRR, compared to that of ConE (10 out of 12 metrics), which suggests an improvement in the ability of zero-shot learning for these queries (please see Appendix E.1 for error bars of the main results).

**Negation queries** Overall, the average MRR in negation queries (AVG_n) of SConE is significantly higher than that of ConE by closely 14% using the FB15k-237 dataset (see Table 1); even though there is no difference in the modeling of negation operator in both models. This can be due to the effect of using RPN to enrich learning in the atomic query structure (1p), which dominantly involves in all negation queries (see Fig. 6 Bottom-Left and Sec. 5.3 for further ablation study in the RPN).

### Table 1: The average MRR (%) of geometric-based embedding models in FOL queries: AVG_p is for EPFO queries while AVG_n is for negation queries. The results of (GQE, Q2B, ConE) are taken from (Zhang et al., 2021). Union queries (2u/up) are in DNF forms, (ip/pi/2u/up) queries are not involved in the training process.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Model</th>
<th>AVG_p</th>
<th>AVG_n</th>
<th>1p</th>
<th>2p</th>
<th>3p</th>
<th>2i</th>
<th>3i</th>
<th>(ip)</th>
<th>(pi)</th>
<th>(2u)</th>
<th>(up)</th>
<th>2in</th>
<th>3in</th>
<th>inp</th>
<th>pin</th>
<th>pci</th>
<th>pni</th>
</tr>
</thead>
<tbody>
<tr>
<td>FB15k</td>
<td>GQE</td>
<td>28.2</td>
<td>-</td>
<td>53.9</td>
<td>15.5</td>
<td>11.1</td>
<td>40.2</td>
<td>52.4</td>
<td>19.4</td>
<td>27.5</td>
<td>22.3</td>
<td>11.7</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Q2B</td>
<td>40.1</td>
<td>-</td>
<td>70.5</td>
<td>23.0</td>
<td>15.1</td>
<td>61.2</td>
<td>71.8</td>
<td>28.7</td>
<td>41.8</td>
<td>37.7</td>
<td>19.0</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Q2P</td>
<td>46.8</td>
<td>16.4</td>
<td>82.6</td>
<td>30.8</td>
<td>25.5</td>
<td>65.1</td>
<td>74.7</td>
<td>34.9</td>
<td>49.5</td>
<td>32.1</td>
<td>26.2</td>
<td>21.9</td>
<td>20.8</td>
<td>12.5</td>
<td>8.9</td>
<td>17.1</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ConE</td>
<td>49.8</td>
<td>14.8</td>
<td>73.3</td>
<td>33.8</td>
<td>29.2</td>
<td>64.4</td>
<td>73.7</td>
<td>35.7</td>
<td>50.9</td>
<td>55.7</td>
<td>31.4</td>
<td>17.9</td>
<td>18.7</td>
<td>12.5</td>
<td>9.8</td>
<td>15.1</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SConE</td>
<td>53.0</td>
<td>16.0</td>
<td>80.8</td>
<td>38.2</td>
<td>30.7</td>
<td>67.0</td>
<td>75.1</td>
<td>41.7</td>
<td>52.1</td>
<td>57.1</td>
<td>34.6</td>
<td>20.5</td>
<td>19.5</td>
<td>14.5</td>
<td>9.2</td>
<td>16.1</td>
<td></td>
</tr>
</tbody>
</table>

| FB15k (237) | GQE | 16.6  | -     | 35.2| 7.4 | 5.5 | 23.6| 35.7| 10.9 | 16.7 | 8.4  | 5.8  | -   | -   | -   | -   |
|             | Q2B | 21.1  | -     | 41.3| 9.9 | 7.2 | 31.1| 45.4| 13.3 | 21.9 | 11.9 | 8.1  | -   | -   | -   | -   |
|             | Q2P | 21.9  | 6.0   | -   | -   | -   | -   | -   | -   | -   | -   | 4.4  | 9.7 | 7.5 | 4.6 | 3.8 |
|             | ConE| 23.4  | 5.9   | 41.8| 12.8| 11.0| 32.6| 47.3| 14.0 | 25.5 | 14.5 | 10.8 | 5.4 | 8.6 | 7.8 | 4.0 |
|             | SConE| 24.1 | 6.7   | 44.2| 13.0| 10.7| 33.8| 47.0| 17.0 | 25.1 | 15.5 | 10.7 | 6.9 | 10.6| 7.9 | 4.3 |

| NELL (995) | GQE | 18.7  | -     | 33.1| 12.1| 9.9 | 27.3| 35.1| 14.5 | 18.5 | 8.5  | 9.0  | -   | -   | -   | -   |
|            | Q2B | 23.6  | -     | 42.7| 14.5| 11.7| 34.7| 45.8| 17.4 | 23.2 | 12.0 | 10.7 | -   | -   | -   | -   |
|            | Q2P | 25.5  | 6.0   | -   | -   | -   | -   | -   | -   | -   | -   | 5.1  | 7.4 | 10.2| 3.3 | 3.4 |
|            | ConE| 27.2  | 6.4   | 53.1| 16.1| 13.9| 40.0| 50.8| 17.5 | 26.3 | 15.3 | 11.3 | 5.7 | 8.1 | 10.8| 3.5 |
|            | SConE| 30.4 | 6.7   | 58.2| 20.5| 17.0| 41.8| 50.7| 22.9 | 28.6 | 18.8 | 15.5 | 6.2 | 8.0 | 11.8| 3.5 |

Figure 6: (Left): training queries. (Left) & (Right): those queries are involved in the evaluation process.
5.3 Ablation study - Sensitivity analysis

We conduct experiments for ablation study w.r.t. two situations: relation projection networks and geometric intersection types; and for sensitivity analysis w.r.t. two hyper-parameters: distance weight ($\psi$) and embedding dimensions ($d$) as follows:

**Relation projection network** Table 2 shows the average MRR on FOL queries in general and several specific EPFO queries (e.g. 1p/2i/ip/2u) of the test set w.r.t. different RPNs: (1) MLP only, (2) relation transformation only and (3) MLP with relation transformation and attention mechanism for the entanglement layer (see Sec. 4.1). Overall, the third scenario observes the highest model’s performance of answering complex logical queries over the other scenarios. We attribute this observation to the advantage of enhancing the learning for atomic query (1p) using the RPN, resulting in an improvement in model performance in total. Specifically, an increase from 42.5 (using MLP only) to 44.2 (using both MLP and relation transformation) in the average MRR (%) of 1p query can lead to an increase in that of other query structures (2i, ip, 2u). This is because the atomic query involves in all structures and in the early stage during the decomposition process via the computation graph for each structure (see Fig. 6). Moreover, SConE with MLP only ($d = 800$, around 20.0M parameters) and symbolic modeling for logical operators uses less parameters than those in ConE ($d = 800$, around 23.9M parameters reported in Long et al. (2022) Appendix B), but both models achieve similar performance (see Table 1).

<table>
<thead>
<tr>
<th>Projection</th>
<th>AVG$_p$</th>
<th>AVG$_s$</th>
<th>1p</th>
<th>2i</th>
<th>ip</th>
<th>2u</th>
<th>#Params</th>
</tr>
</thead>
<tbody>
<tr>
<td>MLP only</td>
<td>22.4</td>
<td>6.5</td>
<td>42.5</td>
<td>30.9</td>
<td>14.5</td>
<td>14.6</td>
<td>11.3M</td>
</tr>
<tr>
<td>MLP only (*)</td>
<td>23.2</td>
<td>6.8</td>
<td>43.1</td>
<td>31.9</td>
<td>16.0</td>
<td>15.1</td>
<td>20.0M</td>
</tr>
<tr>
<td>Rtrans only</td>
<td>23.1</td>
<td>6.5</td>
<td>44.0</td>
<td>32.3</td>
<td>15.7</td>
<td>14.9</td>
<td>25.0M</td>
</tr>
<tr>
<td>Rtrans + MLP + Attention</td>
<td>24.1</td>
<td>6.6</td>
<td>44.2</td>
<td>33.6</td>
<td>17.0</td>
<td>15.3</td>
<td>31.2M</td>
</tr>
</tbody>
</table>

Table 2: Effect of projection network on average MRR (%) fixing $d = 400$ using the FB15k-237 dataset. (*) is for $d = 800$, (M) is million.

**Geometric intersection of sector-cones** Table 3 shows the model’s performance of answering complex logical queries using different types of intersection between sector-cones. The implementation of using individual intersection type is to see the impact of each intersection type on model performance, compared with that using all conditional intersection types. Geometrically, there are three types (None, Complete, Partial) of intersection for any pair of two conjunctive queries as illustrated in Figure 4. Using one type of intersection calculation individually will result in misrepresentation for the other two. For example, assuming all query pairs have None intersection, i.e. only using ($c_3$) and ($c_6$) (see Eq. (4.2) for intersection calculation, we will miss the opportunity to capture Complete and Partial overlap correctly. Table 3 confirms our intuition, the model performs best when all intersection types are considered. Notice that the calculation of intersection of sector-cones neglects the neural-based approach, but the model is able to efficiently learn to answer intersection queries as that in ConE, particularly when using partial intersection only. It is arguable that the learning process focuses on the atomic query which are also involved in conjunctive queries.

<table>
<thead>
<tr>
<th>Intersection</th>
<th>AVG$_p$</th>
<th>AVG$_s$</th>
<th>2i</th>
<th>3i</th>
<th>ip</th>
<th>pi</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>$c_3$, $c_6$</td>
<td>17.6</td>
<td>4.3</td>
<td>20.4</td>
<td>29.8</td>
<td>10.3</td>
</tr>
<tr>
<td>Complete</td>
<td>$c_2$, $c_5$</td>
<td>22.8</td>
<td>6.2</td>
<td>32.0</td>
<td>45.4</td>
<td>14.9</td>
</tr>
<tr>
<td>Partial</td>
<td>$c_1$, $c_4$</td>
<td>23.6</td>
<td>5.6</td>
<td>33.5</td>
<td>48.0</td>
<td>15.5</td>
</tr>
<tr>
<td>All</td>
<td>$c_1$, $c_2$, $c_3$, $c_4$, $c_5$, $c_6$</td>
<td>24.1</td>
<td>6.6</td>
<td>33.6</td>
<td>47.3</td>
<td>17.0</td>
</tr>
</tbody>
</table>

Table 3: Effect of intersection types for sector-cones on average MRR (%) (see Eq. (4.2)) using the FB15k-237 dataset.

**Weight distance** Table 4 (Left) shows the average MRR on FOL queries of the test set w.r.t. different weights ($\psi$) of distances (see Eq. (4.4)), ranging from zero (no axis distance but having inside and outside distance), half of one (equally having axis distance with inside and outside distance) to one (having axis distance but no inside and no outside distance). The model performance increases from ($\psi = 0$) to ($\psi = 1$), which suggests that there is an effect of using the axis distance ($d_a$). The performance reaches its peak when ($\psi$) is set to one as maximum, which suggests that the model can learn to answer complex logical queries using the axis distance only (without inside and outside distance). However, we theorize that the inside and outside distance should be involved during the training process. This is to improve the explainability of cone embeddings, where those entities inside the sector-cone are expected to be answers of the query. In this situation, the aperture plays a role in covering answer entities. Thus, to keep all distance types during the optimization process,
we set $0 < \psi = 0.9 < 1$ for the main results (see Table 1).

Table 4: (Left:) Effect of weight distance ($\psi$) (fixing $d = 400$) and (Right:) Effect of embedding dimension ($d$) (fixing $\psi = 0.9$) on average MRR (%) using the FB15k-237 dataset. (M) is million.

<table>
<thead>
<tr>
<th>SConE</th>
<th>AVG_p</th>
<th>AVG_m</th>
<th>#Params</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\psi = 0.0$</td>
<td>23.1</td>
<td>5.5</td>
<td>4.5M</td>
</tr>
<tr>
<td>$\psi = 0.1$</td>
<td>23.2</td>
<td>5.6</td>
<td>7.4M</td>
</tr>
<tr>
<td>$\psi = 0.5$</td>
<td>23.9</td>
<td>6.0</td>
<td>18.3M</td>
</tr>
<tr>
<td>$\psi = 0.9$</td>
<td>24.1</td>
<td>6.6</td>
<td>31.2M</td>
</tr>
<tr>
<td>$\psi = 1.0$</td>
<td>24.1</td>
<td>6.7</td>
<td>46.3M</td>
</tr>
</tbody>
</table>

Embedding dimension Table 4 (Right) shows the average MRR on FOL queries of the test set w.r.t. different embedding dimension. The model’s performance increases from using small ($d = 64$) to medium ($d = 512$) embedding dimension. This observation suggests that there is a significant effect of this hyper-parameter on the average MRR (of both EPFO and negation queries). Additionally, in the case $d = 256$ (with around 16.3M parameters), SConE uses less 30% in the number of parameters than those in ConE ($d = 800$ with around 23.9M parameters), but both achieves similar average MRR using the same FB15k-237 dataset.

6 Conclusions

We have provided a symbolic modeling for logical operators, particularly computing geometric intersection of sector-cones for modeling the conjunction. In addition, we highlighted the importance of the projection operator by introducing a projection network using neural-based approach, to strengthen the learning in atomic queries involved in all FOL query syntaxes. Our neural-symbolic approach using geometric embeddings significantly outperforms state-of-the-art geometric-based models in both EPFO and negation queries.

Limitations

Although our geometric embedding approach can handle a complete set of basic FOL operators (existential quantification, conjunction, disjunction and negation), the modeling of negation operator cannot narrow down the predicted answers to relevant topics of atomic queries. For example, one can expect the answers of this negation question/query “List Argentina players who are not Lionel Messi in World Cup 2022?” to be any teammates of Lionel Messi (i.e. 2in query structure). However, the current model is designed to return all elements in the entire entity set except for Lionel Messi, which have redundant objects (e.g. trees, music, houses). This is a common limitation not only in geometric-based models but in others using fuzzy sets representation. This is due to the fact that the modeling of negation operator is assumed to be the complement set of a questionable entity w.r.t. the entire entity set. Our hypothesis is that the expected answers should be narrowed into the complement set w.r.t. a sub-topic of relevant entity set.

In addition, when apertures of two sector-cones are obtuse angles, the current calculation of partial intersection cannot correctly model the conjunction operator. This special case is inevitable in a system using geometric representation that is closed under negation and conjunction, but not for disjunction (see Appendix A.2 for further details).

Ethics Statement

The ability of models to answer complex logical queries is achievable to reason about knowledge graphs. Due to model’s uncertainty, one potential negative impact of this task is the out-of-control in automatic reasoning over open large-scale knowledge graphs, where there are diverse source of information. Some of which though can be missed from KGs due to incompleteness or private purposes, but they can be possibly reasoned using the query embedding methods.

Acknowledgements

This research is supported by the Australian Research Council through the Centre for Transforming Maintenance through Data Science (grant number IC180100030), funded by the Australian Government. Wei Liu acknowledges the support from ARC Discovery Projects DP150102405. Further, the authors would like to thank all the anonymous reviewers for their insightful feedback.

References


A Modeling logical operators - None
intersection type

A.1 Comments on partial and complete type of intersection

Note that the condition \((c_1)\) for partial intersection type has a special case when these equalities holds \((u_2 = l_1)\) and \((l_1 = l_2)\). In this situation, the partial type \((c_1)\) can be considered as the complete type \((c_2)\). Hence, these types of intersection can be used interchangeably. Similarly, we can interchangeably use the calculation of the partial and complete type for the condition \((c_4, c_5)\). This is due to the fact that the calculation of these intersection types become as follows:

\[
\beta_\land = \begin{cases} 
    0, & \text{if } c_1, \\
    \alpha_2, & \text{if } c_2, \\
    0, & \text{if } c_4, \\
    \alpha_1, & \text{if } c_5.
\end{cases}
\]

In terms of the cases \((c_1, c_2)\), notice that \((u_2 = l_1 = l_2)\) or \((u_2 = l_2)\), hence \((0 = \beta_2)\), and \((u_2 = \alpha_2 + \frac{\beta_2}{2})\) or \((u_2 = \alpha_2)\). Similar explanation is for the cases \((c_4, c_5)\).

A.2 Special case of partial intersection

In case of the aperture of two sector-cones are both obtuse angles as shown in Fig. 7, the calculation of partial intersection under conditions \((c_1, c_4)\) cannot exactly model the conjunction operator. Specially, the calculation of \((\alpha_\land, \beta_\land)\) for conjunction operator in Sec. 4.2 will return \((\alpha_\land, \beta_\land)\) of the right intersection sector-cone, but will ignore \((\alpha_\land, \beta_\land)\) of the left intersection sector-cone in this special case. Our hypothesis is that this special case is inevitable if a system using geometric representation, such as conic shapes, is closed under negation and conjunction, but not disjunction. This limitation can be addressed by providing a refined approach for calculating the partial intersection when both apertures are obtuse angles. We leave this approach as an extension for future work.

B Further details in section modeling operators of FOL queries

B.1 Scaling function

Continuing Sec. 4.1, after obtaining the output from entanglement layer as \(q = (\alpha, \beta)\), we scale the semantic center axis and the aperture into their normal ranges \([-\pi, \pi]\) and \([0, 2\pi]\) respectively, defined in Sec. 3.2. The final embedded query \(q = (\alpha', \beta')\) is as follows:

\[
\alpha' = \pi \tanh(\lambda_1 \alpha), \quad \beta' = \pi \tanh(\lambda_2 \beta + \pi),
\]

where \((\lambda_1, \lambda_2)\) are scaling hyper-parameters.

<table>
<thead>
<tr>
<th>Stats</th>
<th>FB15k</th>
<th>FB15k-237</th>
<th>NELL995</th>
</tr>
</thead>
<tbody>
<tr>
<td>Entities</td>
<td>14,951</td>
<td>14,505</td>
<td>63,361</td>
</tr>
<tr>
<td>Relations</td>
<td>1,345</td>
<td>237</td>
<td>200</td>
</tr>
<tr>
<td>Triples Valid</td>
<td>50,000</td>
<td>17,526</td>
<td>114,213</td>
</tr>
<tr>
<td>(Edges) Test</td>
<td>59,097</td>
<td>20,101</td>
<td>14,324</td>
</tr>
<tr>
<td>Total</td>
<td>592,213</td>
<td>310,079</td>
<td>142,804</td>
</tr>
</tbody>
</table>

Table 5: Statistics of three datasets, reported from Ren and Leskovec (2020)

<table>
<thead>
<tr>
<th>Split</th>
<th>Query syntaxes</th>
<th>FB15k</th>
<th>FB15k-237</th>
<th>NELL995</th>
</tr>
</thead>
<tbody>
<tr>
<td>Train 1p/2p/3p/2i/3i</td>
<td>273,710</td>
<td>149,689</td>
<td>107,982</td>
<td></td>
</tr>
<tr>
<td>2in/3in/2i/3i</td>
<td>27,371</td>
<td>14,968</td>
<td>10,798</td>
<td></td>
</tr>
<tr>
<td>Valid 1p</td>
<td>59,097</td>
<td>20,101</td>
<td>16,927</td>
<td></td>
</tr>
<tr>
<td>Others (Each)</td>
<td>8,000</td>
<td>5,000</td>
<td>4,000</td>
<td></td>
</tr>
<tr>
<td>Test 1p</td>
<td>67,016</td>
<td>22,812</td>
<td>17,034</td>
<td></td>
</tr>
<tr>
<td>Others (Each)</td>
<td>8,000</td>
<td>5,000</td>
<td>4,000</td>
<td></td>
</tr>
</tbody>
</table>

Table 6: Statistics of query structures preprocessed by Ren and Leskovec (2020).

C Distance score functions

Continuing the calculation of distance score function in Sec. 4.3, the calculations of the outside with inside distances and axis distance are as follows:

\[
d_o = \left\| \min \{d_l, d_u\} \right\|_1, \\
d_i = \left\| \min \{d_\alpha, d_\beta\} \right\|_1, \\
d_a = \left\| \alpha - \alpha_q \right\|_1,
\]

where \(||·||_1\) denotes the \(L_1\) norm, the upper bound \((u = \alpha_q + \frac{\beta_2}{2})\) and the lower bound \((l = \alpha_q - \frac{\beta_2}{2})\) are of the query \((q)\): \(d_l = |1 - \cos(\alpha - 1)|\) and \(d_u = |1 - \cos(\alpha - u)|\) is the lower and upper bound outside distance respectively, \(d_a = |1 - \cos(\alpha - \alpha_q)|\) and \(d_\beta = |1 - \cos(\beta_2)|\) is the axis and the aperture inside distance respectively.
Table 7: Found hyper-parameters for the main results.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>d</th>
<th>b</th>
<th>n</th>
<th>m</th>
<th>γ</th>
<th>l</th>
<th>ψ</th>
<th>λ1</th>
<th>λ2</th>
<th>λ</th>
</tr>
</thead>
<tbody>
<tr>
<td>FB15k</td>
<td>400</td>
<td>512</td>
<td>128</td>
<td>450k</td>
<td>30</td>
<td>0.00005</td>
<td>0.9</td>
<td>1.0</td>
<td>2.0</td>
<td>0.02</td>
</tr>
<tr>
<td>FB15k-237</td>
<td>400</td>
<td>512</td>
<td>128</td>
<td>350k</td>
<td>20</td>
<td>0.00005</td>
<td>0.9</td>
<td>1.0</td>
<td>2.0</td>
<td>0.02</td>
</tr>
<tr>
<td>NELL995</td>
<td>400</td>
<td>512</td>
<td>128</td>
<td>350k</td>
<td>20</td>
<td>0.00005</td>
<td>0.9</td>
<td>1.0</td>
<td>2.0</td>
<td>0.02</td>
</tr>
</tbody>
</table>

Table 8: Error bars of MRR (%) for the main results of SConE. (±) is for standard deviation.

D Experimental setups

D.1 Datasets

Following experimental settings in Ren and Leskovec (2020) for the training and evaluation process, which pre-processed datasets (FB15k Bolbacker et al. (2008), FB15k-237 Toutanova and Chen (2015) and NELL995 Xiong et al. (2017)) and publicly available at this link ³, Table 5 shows statistics of these datasets regarding the number of entities, the number of relations and the number of triples. In addition, Table 6 shows the number of queries in different structures for the training/validation/test set.

D.2 Training and evaluation settings:

Further details, hyper-parameters and error bars

Following the original work in (Ren and Leskovec, 2020) ⁴, we implement all experiments using Pytorch as Deep Learning framework under Python. For each experiment, we conduct it on a single NVIDIA Tesla V100 GPU, in the UWA Kaya High Performance Computing (HPC) cluster. For the settings in the relation projection network, we use a three-layer MLP using 1600 dimension for hidden layers and Rectified Linear Units (ReLU) for the activation function. Further, following the found hyper-parameters in ConE (Zhang et al., 2021): \( \lambda_1 = 1.0, \lambda_2 = 2.0, \lambda = 0.02, \) batch size \( b = 512 \) and negative sampling size \( n = 128 \), we use these in all experiments. With regard to other hyper-parameters, we search for the best performance in MRR. Specifically, \((\gamma)\) involving in the loss function is in \( \{20, 30\} \), the learning rate \((l)\) is in \((1e^{-4}, 5e^{-5})\). Table 7 shows found hyper-parameters of the main results in Table 1. For obtaining error bars of the main results, we run the model five times, each uses different random seed in \( \{0, 10, 100, 1000, 10000\} \) (see Table 8 for further details).

E Additional results

E.1 Error bars for the main results

Table 8 shows error bars of the average MRR (in percentage) for the main results of SConE reported in Table 1 (see random seed settings as described in Appendix. D.2). We compute the standard deviation (std) of results from five experiments using each of the three dataset (FB15k, FB15k-237 and NELL995). Overall, the error bar of the average MRR is low in all query structures and in average for EPFO and negation queries, which demonstrates the stability of model performance.

E.2 Comparison results with fuzzy logic-based models

Table 9 shows comparison in the average MRR (in percentage) of SConE with that of other fuzzy logic-based models. In the FB15k-237 dataset, GNN-QE achieves state-of-the-art performance of answering complex logical queries. Compared to other models (ENeSy and FuzzQE), the performance of SConE is nearly to that of these logic-based models. With regard to the NELL995 dataset, although SConE achieves the lowest performance in answering negation queries, the performance of SConE reaches its highest in answering non-
negation queries among other logic-based models. Particularly, there is a significant improvement in the average MRR regarding union queries, compared to that in other models.

### F Further sensitivity analysis - Weight of semantic axis for none intersection type

Table 10 shows the performance of SConE w.r.t. different weights \((\delta \in [0, 1])\) of semantic axis in the case of none type intersection (see Eq. (4.3)). We conduct five experiments using different weights \(\{0.0, 0.1, 0.5, 0.9, 1.0\}\). When \(\delta = 0.0\) or \(\delta = 1.0\), the semantic axis of an intersection query corresponds to the lower bound of one sector-cone or the upper bound of the other sector-cone. When \(\delta = 0.1\) or \(\delta = 0.9\), the spatial position of this semantic axis is close the lower bound of one sector-cone or the upper bound of the other sector-cone respectively. In a special case when \(\delta = 0.5\), the semantic axis is in the middle of the two bounds.

Overall, the average MRR (AVG) of SConE for both non-negation and negation queries is similar from one to another in all different weights \(\delta\). However, there is a slight difference between \(\text{AVG}_p\) for non-negation queries and \(\text{AVG}_n\) for negation queries. When \(\delta = 0.1\), SConE achieves the highest \(\text{AVG}_p\) but not for \(\text{AVG}_n\). In contrast, when \(\delta = 0.5\), SConE achieves the highest \(\text{AVG}_n\) but not for \(\text{AVG}_p\). Since there is a slight difference in \(\text{AVG}_p\) using \(\delta = 0.1\) and \(\delta = 0.5\) but there is highly difference in \(\text{AVG}_n\) using these weights, we select the special case with \(\delta = 0.5\) or the middle semantic axis of intersection query and report the main results. Further, we observe that there is no significant difference in the average MRR (AVG) of model performance for both non-negation and negation queries (see second column of Table 10). Thus, any semantic axes (or cones) between the two mentioned bounds can be considered as the semantic axis of intersection query. Note that the aperture of intersection query in the case of none intersection type is equivalent to zero.

### Table 9: Comparison the average MRR (%) of SConE with that of logic-based embedding models (CQD-CO, CQD-Beam, FuzzQE, ENeSy, GNN-QE). Union queries (2u/up) are in DNF forms. Results of CQD-CO, CQD-Beam, GNN-QE are taken from (Zhu et al., 2022).

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Model</th>
<th>(\text{AVG}_p)</th>
<th>(\text{AVG}_n)</th>
<th>1p</th>
<th>2p</th>
<th>3p</th>
<th>2i</th>
<th>3i</th>
<th>ip</th>
<th>pi</th>
<th>2u</th>
<th>up</th>
<th>2in</th>
<th>3in</th>
<th>inp</th>
<th>pin</th>
<th>pni</th>
</tr>
</thead>
<tbody>
<tr>
<td>FB15k-237</td>
<td>CQD-CO</td>
<td>21.8</td>
<td>46.7</td>
<td>9.5</td>
<td>6.3</td>
<td>31.2</td>
<td>40.6</td>
<td>16.0</td>
<td>23.6</td>
<td>14.5</td>
<td>8.2</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>CQD-Beam</td>
<td>22.3</td>
<td>46.7</td>
<td>11.6</td>
<td>8.0</td>
<td>31.2</td>
<td>40.6</td>
<td>18.7</td>
<td>21.2</td>
<td>14.6</td>
<td>8.4</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>FuzzQE</td>
<td>24.2</td>
<td>8.5</td>
<td>42.2</td>
<td>13.3</td>
<td>10.2</td>
<td>33.0</td>
<td>47.3</td>
<td>18.9</td>
<td>26.2</td>
<td>15.6</td>
<td>10.8</td>
<td>9.7</td>
<td>12.6</td>
<td>7.8</td>
<td>5.8</td>
<td>6.6</td>
</tr>
<tr>
<td></td>
<td>ENeSy</td>
<td>24.5</td>
<td>8.5</td>
<td>44.7</td>
<td>11.7</td>
<td>8.6</td>
<td>34.8</td>
<td>50.4</td>
<td>19.7</td>
<td>27.6</td>
<td>14.2</td>
<td>8.4</td>
<td>10.1</td>
<td>10.4</td>
<td>7.6</td>
<td>6.1</td>
<td>8.1</td>
</tr>
<tr>
<td></td>
<td>GNN-QE</td>
<td>26.8</td>
<td>10.2</td>
<td>42.8</td>
<td>14.7</td>
<td>11.8</td>
<td>38.3</td>
<td>54.1</td>
<td>18.9</td>
<td>31.1</td>
<td>16.2</td>
<td>13.4</td>
<td>10.0</td>
<td>16.8</td>
<td>9.3</td>
<td>7.2</td>
<td>7.8</td>
</tr>
<tr>
<td></td>
<td>SConE</td>
<td>24.1</td>
<td>6.7</td>
<td>44.2</td>
<td>13.0</td>
<td>10.7</td>
<td>33.8</td>
<td>47.0</td>
<td>17.0</td>
<td>25.1</td>
<td>15.5</td>
<td>10.7</td>
<td>6.9</td>
<td>10.6</td>
<td>7.9</td>
<td>4.0</td>
<td>4.3</td>
</tr>
</tbody>
</table>

| NELL995 | CQD-CO   | 28.8             | 60.4             | 17.8 | 12.7 | 39.3 | 46.6 | 22.0 | 30.1 | 17.3 | 13.2 | -   | -   | -   | -   | -   | -   |
|         | CQD-Beam | 28.6             | 60.4             | 20.6 | 11.6 | 39.3 | 46.6 | 23.9 | 25.4 | 17.5 | 12.2 | -   | -   | -   | -   | -   | -   |
|         | FuzzQE   | 28.9             | 9.7              | 53.3 | 18.9 | 14.9 | 42.4 | 52.5 | 18.9 | 30.8 | 15.9 | 12.6 | 9.9 | 14.6 | 11.4 | 6.3 | 6.3 |
|         | ENeSy    | 29.3             | 8.0              | 58.1 | 19.3 | 15.7 | 39.8 | 50.3 | 21.8 | 28.1 | 17.3 | 13.7 | 8.3 | 10.2 | 11.5 | 4.6 | 5.4 |
|         | GNN-QE   | 29.4             | 9.8              | 59.0 | 18.0 | 14.0 | 39.6 | 49.8 | 24.8 | 29.8 | 16.4 | 13.1 | 11.3 | 8.5 | 11.6 | 8.6 | 8.8 |
|         | SConE    | 30.4             | 6.7              | 58.2 | 20.5 | 17.0 | 41.8 | 50.7 | 22.9 | 28.6 | 18.8 | 15.5 | 6.2 | 8.0 | 11.8 | 3.5 | 4.2 |

<table>
<thead>
<tr>
<th>Delta</th>
<th>AVG(_p)</th>
<th>AVG(_n)</th>
<th>2i</th>
<th>3i</th>
<th>ip</th>
<th>pi</th>
<th>2in</th>
<th>3in</th>
<th>inp</th>
<th>pin</th>
<th>pni</th>
</tr>
</thead>
<tbody>
<tr>
<td>(\delta = 0.0)</td>
<td>17.8</td>
<td>24.4</td>
<td>5.9</td>
<td>34.5</td>
<td>48.6</td>
<td>16.8</td>
<td>26.6</td>
<td>5.3</td>
<td>9.3</td>
<td>7.9</td>
<td>3.6</td>
</tr>
<tr>
<td>(\delta = 0.1)</td>
<td>17.9</td>
<td>24.4</td>
<td>6.1</td>
<td>34.3</td>
<td>48.2</td>
<td>17.0</td>
<td>26.2</td>
<td>5.6</td>
<td>9.7</td>
<td>7.8</td>
<td>3.8</td>
</tr>
<tr>
<td>(\delta = 0.5)</td>
<td>17.9</td>
<td>24.1</td>
<td>6.8</td>
<td>33.9</td>
<td>47.0</td>
<td>16.9</td>
<td>25.1</td>
<td>6.9</td>
<td>10.8</td>
<td>7.9</td>
<td>4.0</td>
</tr>
<tr>
<td>(\delta = 0.9)</td>
<td>17.8</td>
<td>24.3</td>
<td>6.2</td>
<td>33.9</td>
<td>47.7</td>
<td>16.8</td>
<td>26.1</td>
<td>6.0</td>
<td>9.8</td>
<td>7.8</td>
<td>3.8</td>
</tr>
<tr>
<td>(\delta = 1.0)</td>
<td>17.8</td>
<td>24.3</td>
<td>6.1</td>
<td>34.1</td>
<td>47.6</td>
<td>16.8</td>
<td>26.3</td>
<td>5.8</td>
<td>9.7</td>
<td>7.8</td>
<td>3.7</td>
</tr>
</tbody>
</table>

Table 10: The effect of weight axis \(\delta\) for the none type of intersection queries on average MRR (%) (see Eq. (4.3)).
ACL 2023 Responsible NLP Checklist

A For every submission:

✓ A1. Did you describe the limitations of your work?
   *Section Limitations (after Section 6 Conclusion), Appendix A.3*

✓ A2. Did you discuss any potential risks of your work?
   *Section Ethics Statement (after Section Limitations)*

✓ A3. Do the abstract and introduction summarize the paper’s main claims?
   *Section Abstract and Section 1*

✘ A4. Have you used AI writing assistants when working on this paper?
   *Left blank.*

B ❊ Did you use or create scientific artifacts?
   *Left blank.*

☐ B1. Did you cite the creators of artifacts you used?
   *No response.*

☐ B2. Did you discuss the license or terms for use and / or distribution of any artifacts?
   *No response.*

☐ B3. Did you discuss if your use of existing artifact(s) was consistent with their intended use, provided that it was specified? For the artifacts you create, do you specify intended use and whether that is compatible with the original access conditions (in particular, derivatives of data accessed for research purposes should not be used outside of research contexts)?
   *No response.*

☐ B4. Did you discuss the steps taken to check whether the data that was collected / used contains any information that names or uniquely identifies individual people or offensive content, and the steps taken to protect / anonymize it?
   *No response.*

☐ B5. Did you provide documentation of the artifacts, e.g., coverage of domains, languages, and linguistic phenomena, demographic groups represented, etc.?
   *No response.*

☐ B6. Did you report relevant statistics like the number of examples, details of train / test / dev splits, etc. for the data that you used / created? Even for commonly-used benchmark datasets, include the number of examples in train / validation / test splits, as these provide necessary context for a reader to understand experimental results. For example, small differences in accuracy on large test sets may be significant, while on small test sets they may not be.
   *No response.*

C ✓ Did you run computational experiments?
   *Section 5*

✓ C1. Did you report the number of parameters in the models used, the total computational budget (e.g., GPU hours), and computing infrastructure used?
   *Section 5.3, Appendix D.2*

The Responsible NLP Checklist used at ACL 2023 is adopted from NAACL 2022, with the addition of a question on AI writing assistance.
C2. Did you discuss the experimental setup, including hyperparameter search and best-found hyperparameter values?
   Section 5.1, Appendix D.1, D.2

C3. Did you report descriptive statistics about your results (e.g., error bars around results, summary statistics from sets of experiments), and is it transparent whether you are reporting the max, mean, etc. or just a single run?
   Appendix D.2, Appendix E.1

☐ C4. If you used existing packages (e.g., for preprocessing, for normalization, or for evaluation), did you report the implementation, model, and parameter settings used (e.g., NLTK, Spacy, ROUGE, etc.)?
   Not applicable. Appendix D.2

D  ☒ Did you use human annotators (e.g., crowdworkers) or research with human participants?
   Left blank.

☐ D1. Did you report the full text of instructions given to participants, including e.g., screenshots, disclaimers of any risks to participants or annotators, etc.?
   No response.

☐ D2. Did you report information about how you recruited (e.g., crowdsourcing platform, students) and paid participants, and discuss if such payment is adequate given the participants’ demographic (e.g., country of residence)?
   No response.

☐ D3. Did you discuss whether and how consent was obtained from people whose data you’re using/curating? For example, if you collected data via crowdsourcing, did your instructions to crowdworkers explain how the data would be used?
   No response.

☐ D4. Was the data collection protocol approved (or determined exempt) by an ethics review board?
   No response.

☐ D5. Did you report the basic demographic and geographic characteristics of the annotator population that is the source of the data?
   No response.